```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBBBBBBBBBBB       AAA       AAA     SSSSSSSSS   RRRRRRRRRRR           TTT        LLL
BBBBBBBBBBBB       AAA       AAA     SSSSSSSSS   RRRRRRRRRRR           TTT        LLL
BBBBBBBBBBBB       AAA       AAA     SSSSSSSSS   RRRRRRRRRRR           TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR             TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR             TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR             TTT        LLL
BBB          BBB   AAA       AAA           SSS  RRR      RRR           TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR             TTT        LLL
BBB          BBB   AAA       AAA           SSS  RRR      RRR           TTT        LLL
BBB          BBB   AAA       AAA           SSS  RRR      RRR           TTT        LLL
BBB          BBB   AAA       AAA           SSS  RRR      RRR           TTT        LLL
BBBBBBBBBBBB       AAA       AAA   SSSSSSSSSSS   RRR         RRR       TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA       AAA   SSSSSSSSSSS   RRR         RRR       TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA       AAA   SSSSSSSSSSS   RRR         RRR       TTT        LLLLLLLLLLLLLLL
```

```
BBBBBBBB      AAAAAA      SSSSSSSS    CCCCCCCC  TTTTTTTTTT  RRRRRRRR    LL            CCCCCCCC
BBBBBBBB      AAAAAA      SSSSSSSS    CCCCCCCC  TTTTTTTTTT  RRRRRRRR    LL            CCCCCCCC
BB      BB   AA    AA   SS          CC            TT      RR      RR  LL          CC
BB      BB   AA    AA   SS          CC            TT      RR      RR  LL          CC
BB      BB   AA    AA   SS          CC            TT      RR      RR  LL          CC
BBBBBBBB     AA    AA     SSSSSS    CC            TT      RRRRRRRR    LL          CC
BBBBBBBB     AA    AA     SSSSSS    CC            TT      RRRRRRRR    LL          CC
BB      BB   AAAAAAAAAA         SS  CC            TT      RR  RR      LL          CC
BB      BB   AAAAAAAAAA         SS  CC            TT      RR  RR      LL          CC
BB      BB   AA    AA           SS  CC            TT      RR    RR    LL          CC              ....
BB      BB   AA    AA           SS  CC            TT      RR    RR    LL          CC              ....
BBBBBBBB     AA    AA   SSSSSSSS      CCCCCCCC    TT      RR      RR  LLLLLLLLLL    CCCCCCCC    ....
BBBBBBBB     AA    AA   SSSSSSSS      CCCCCCCC    TT      RR      RR  LLLLLLLLLL    CCCCCCCC    ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE BAS$CTRLC (                          ! Control C handler
;     2          0002  0                 IDENT = '2-005'           ! File: BASCTRLC.B32 Edit: MDL2005
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !**************************************************************************
;     7          0007  1 !*                                                                        *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                              *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                  *
;    11          0011  1 !*                                                                        *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    17          0017  1 !*  TRANSFERRED.                                                           *
;    18          0018  1 !*                                                                        *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    21          0021  1 !*  CORPORATION.                                                           *
;    22          0022  1 !*                                                                        *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
;    25          0025  1 !*                                                                        *
;    26          0026  1 !*                                                                        *
;    27          0027  1 !**************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:  VAX-11 BASIC Miscellaneous Support
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains routines for enabling, disabling, and
;    36          0036  1 !       handling Control C interrupts.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  VAX-11 User Mode
;    39          0039  1 !
;    40          0040  1 ! AUTHOR: John Sauter, CREATION DATE: 19-FEB-1979
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! 1-001 - Original.  JBS 19-FEB-1979
;    45          0045  1 ! 1-002 - Add a handler to the AST routine to catch UNWINDS, making
;    46          0046  1 !         sure that they dismiss the AST properly.  JBS 20-FEB-1979
;    47          0047  1 ! 1-003 - Add BAS$$CTRLC_INIT, for the RUN command.  JBS 22-JUN-1979
;    48          0048  1 ! 1-004 - If a control C trap goes off but the user was not enabled,
;    49          0049  1 !         signal an INFO message to the keyboard monitor, who may
;    50          0050  1 !         wish to continue.  JBS 14-SEP-1979
;    51          0051  1 ! 1-005 - Use SYS$INPUT rather than TT.  JBS 20-SEP-1979
;    52          0052  1 ! 1-006 - Call SYS$CLRAST to clear the AST, rather than using CHMK.
;    53          0053  1 !         JBS 27-NOV-1979
;    54          0054  1 ! 1-007 - Do translations of SYS$INPUT until it fails to translate.
;    55          0055  1 !         JBS 24-JUL-1980
;    56          0056  1 ! 1-008 - Clear the AST immediately in CONTROL_C.  PLL 7-Aug-81
;    57          0057  1 ! 1-009 - Use LIB$GET_EF to obtain event flags for $QIOWs.  PLL 30-Nov-81
```

```
;    58          0058  1 ! 1-010 - Don't turn off control c's when a control c AST goes off.
;    59          0059  1 !         They should be turned off only by the RCTRLC function.  PLL 22-Jun-82
;    60          0060  1 ! 1-011 - Edit 010 should also have checked RUN_CMD in CONTROL_C, so that
;    61          0061  1 !         ctrlc's are always enabled in immediate mode from the VMS point of
;    62          0062  1 !         view.  PLL 6-Jul-1982
;    63          0063  1 ! 1-012 - make ERN and ERL available when user hits CTRL/C from inside
;    64          0064  1 !         the environment.  MDL 22-Jul-1982
;    65          0065  1 ! 2-001 - rewrite to use permanent AST enabling.  Also allow CTRLC function
;    66          0066  1 !         to work when program runs from a command procedure.  MDL 28-Sep-1983
;    67          0067  1 ! 2-002 - don't use SYS$CLRAST - it causes us to never return to where we
;    68          0068  1 !         were before the AST occurred.  MDL 4-Jan-1984
;    69          0069  1 ! 2-003 - check if I/O in progress before signalling at AST level, and simply
;    70          0070  1 !         return if so.  add new routine BAS$$SIGNAL_CTRLC for use from REC
;    71          0071  1 !         level I/O routines.  Coordinated change with BAS$$REC_PROC 1-093.
;    72          0072  1 !         MDL 12-Mar-1984
;    73          0073  1 ! 2-004 - RMS will only return RMS$_CONTROLC for an interrupted terminal I/O,
;    74          0074  1 !         therefore we must signal in all other cases.  MDL 3-Apr-1984
;    75          0075  1 ! 2-005 - only signal if we're really enabled.  MDL 10-Apr-1984
;    76          0076  1 !--
;    77          0077  1
;    78          0078  1 !<BLF/PAGE>
```

```
;    80          0079  1 !
;    81          0080  1 ! SWITCHES:
;    82          0081  1 !
;    83          0082  1
;    84          0083  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    85          0084  1
;    86          0085  1 !
;    87          0086  1 ! LINKAGES:
;    88          0087  1 !
;    89          0088  1 !       NONE
;    90          0089  1 !
;    91          0090  1 ! TABLE OF CONTENTS:
;    92          0091  1 !
;    93          0092  1
;    94          0093  1 FORWARD ROUTINE
;    95          0094  1     BAS$CTRLC,                              ! Enable Control C interrupts
;    96          0095  1     BAS$RCTRLC,                             ! Disable Control C interrupts
;    97          0096  1     BAS$$CTRLC_INIT : NOVALUE,              ! Set up for RUN command
;    98          0097  1     BAS$$SIGNAL_CTRLC : NOVALUE,            ! Signal the CTRL/C condition
;    99          0098  1     CONTROL_C : NOVALUE;                    ! Handle a Control C interrupt
;   100          0099  1
;   101          0100  1 !
;   102          0101  1 ! INCLUDE FILES:
;   103          0102  1 !
;   104          0103  1
;   105          0104  1 REQUIRE 'RTLIN:RTLPSECT';                   ! Macros for defining psects
;   106          0199  1
;   107          0200  1 REQUIRE 'RTLIN:BASFRAME';                   ! BASIC frame definitions
;   108          0403  1
;   109          0404  1 REQUIRE 'RTLML:OTSLUB';                     ! LUB definitions
;   110          0544  1
;   111          0545  1 REQUIRE 'RTLIN:OTSLNK';                     ! linkage definitions
;   112          0974  1
;   113          0975  1 LIBRARY 'RTLSTARLE';                        ! Define system symbols
;   114          0976  1
;   115          0977  1 !
;   116          0978  1 ! MACROS:
;   117          0979  1 !
;   118          0980  1 !       NONE
;   119          0981  1 !
;   120          0982  1 ! EQUATED SYMBOLS:
;   121          0983  1 !
;   122          0984  1 !       NONE
;   123          0985  1 !
;   124          0986  1 ! PSECTS:
;   125          0987  1 !
;   126          0988  1 DECLARE_PSECTS (BAS);                       ! Declare psects for BAS$ facility
;   127          0989  1 !
;   128          0990  1 ! OWN STORAGE:
;   129          0991  1 !
;   130          0992  1
;   131          0993  1 OWN
;   132          0994  1     TT_CHAN : UNSIGNED WORD INITIAL (WORD (0)), ! The channel the terminal is assigned on
;   133          0995  1     RUN_CMD : BYTE INITIAL (BYTE (0)),          ! Set if we are in the RUN command
;   134          0996  1     CC_REALLY_ENABLED : BYTE INITIAL (BYTE (0)),! Set if the user has control C traps enabled
;   135          0997  1     CC_ENABLED_USER_PT_OF_VIEW: BYTE INITIAL (BYTE (0));
;   136          0998  1                                                 ! Set if the user thinks he has ctrl/c enabled
```

```
;  137          0999  1
;  138          1000  1 !
;  139          1001  1 ! EXTERNAL REFERENCES:
;  140          1002  1 !
;  141          1003  1
;  142          1004  1 EXTERNAL ROUTINE
;  143          1005  1     LIB$GET_EF,                          ! allocate an event flag
;  144          1006  1     LIB$FREE_EF,                         ! deallocate an event flag
;  145          1007  1     LIB$SIGNAL,                          ! Signal a condition
;  146          1008  1     LIB$STOP : NOVALUE,                  ! Signal a fatal error
;  147          1009  1     LIB$MATCH_COND,                      ! Match condition codes
;  148          1010  1     BAS$$CB_PUSH : JSB_CB_PUSH NOVALUE,  ! Load register CCB
;  149          1011  1     BAS$$CB_POP : JSB_CB_POP NOVALUE,    ! Release register CCB
;  150          1012  1     BAS$$LINE,                           ! get current line
;  151          1013  1     BAS$$MODULE,                         ! get current module name
;  152          1014  1     BAS$HANDLER;                         ! just need address of this
;  153          1015  1
;  154          1016  1 EXTERNAL
;  155          1017  1     BAS$T_ERN : BLOCK [8, BYTE] ,        ! descriptor for module name
;  156          1018  1     BAS$L_ERR ,                          ! current error code
;  157          1019  1     BAS$L_ERL ,                          ! line number of error
;  158          1020  1     OTS$$V_IOINPROG : VOLATILE BITVECTOR;! channels w/ I/O in progress
;  159          1021  1
;  160          1022  1
;  161          1023  1 !+
;  162          1024  1 ! The following are the error codes used in this module.
;  163          1025  1 !-
;  164          1026  1
;  165          1027  1 EXTERNAL LITERAL
;  166          1028  1     BAS$K_PROC__TRA,
;  167          1029  1     BAS$_PROC__TRA;                      ! Programmable ^C trap
;  168          1030  1
```

```
;  170     1031  1 GLOBAL ROUTINE BAS$CTRLC =                    ! Enable Control C interrupts
;  171     1032  1
;  172     1033  1 !++
;  173     1034  1 ! FUNCTIONAL DESCRIPTION:
;  174     1035  1 !
;  175     1036  1 !       Enable Control C traps, so that a Control C will cause the
;  176     1037  1 !       user's program to take an ON ERROR GOTO branch.
;  177     1038  1 !
;  178     1039  1 ! FORMAL PARAMETERS:
;  179     1040  1 !
;  180     1041  1 !       NONE
;  181     1042  1 !
;  182     1043  1 ! IMPLICIT INPUTS:
;  183     1044  1 !
;  184     1045  1 !       NONE
;  185     1046  1 !
;  186     1047  1 ! IMPLICIT OUTPUTS:
;  187     1048  1 !
;  188     1049  1 !       NONE
;  189     1050  1 !
;  190     1051  1 ! ROUTINE VALUE:
;  191     1052  1 !
;  192     1053  1 !       Always returns zero.
;  193     1054  1 !
;  194     1055  1 ! SIDE EFFECTS:
;  195     1056  1 !
;  196     1057  1 !       Leaves Control C traps enabled if the process has a terminal.
;  197     1058  1 !
;  198     1059  1 !--
;  199     1060  1
;  200     1061  2     BEGIN
;  201     1062  2
;  202     1063  2 !+
;  203     1064  2 ! If CTRL/C reception is not currently enabled, begin some investigation.
;  204     1065  2 !-
;  205     1066  3     IF ( NOT .CC_REALLY_ENABLED )
;  206     1067  2     THEN
;  207     1068  3         BEGIN
;  208     1069  3
;  209     1070  3         LOCAL
;  210     1071  3             ASSIGN_RESULT,
;  211     1072  3             QIO_RESULT,
;  212     1073  3             GETDVI_RESULT,
;  213     1074  3             GETJPI_RESULT,
;  214     1075  3             STATUS,
;  215     1076  3
;  216     1077  3             EVENT_FLAG,
;  217     1078  3
;  218     1079  3             CONTROL_CHARS : VECTOR [2, LONG] INITIAL (0, 8),
;  219     1080  3
;  220     1081  3             DEVICE_CLASS : INITIAL(0),
;  221     1082  3             DEVNAM_DESC : BLOCK [8, BYTE],
;  222     1083  3             DVI_RETURN_LENGTH : INITIAL(0),
;  223     1084  3             DVI_ITEMS : VECTOR [4, LONG] INITIAL ( ((DVI$_DEVCLASS^16) OR 4),
;  224     1085  3                                                    DEVICE_CLASS,
;  225     1086  3                                                    DVI_RETURN_LENGTH,
;  226     1087  3                                                    0 ),
```

```
;  227          1088  3
;  228          1089  3                TERMINAL_NAME : VECTOR [256, BYTE],
;  229          1090  3                JPI_RETURN_LENGTH : INITIAL(0),
;  230          1091  3                JPI_ITEMS : VECTOR [4, LONG] INITIAL ( ((JPI$_TERMINAL^16) OR 256),
;  231          1092  3                                                       TERMINAL_NAME,
;  232          1093  3                                                       JPI_RETURN_LENGTH,
;  233          1094  3                                                       0 );
;  234          1095  3
;  235          1096  3            !+
;  236          1097  3            ! see if SYS$INPUT is a terminal device.
;  237          1098  3            !-
;  238          1099  3                    DEVNAM_DESC [DSC$W_LENGTH] = %CHARCOUNT ('SYS$INPUT');
;  239          1100  3                    DEVNAM_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  240          1101  3                    DEVNAM_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  241          1102  3                    DEVNAM_DESC [DSC$A_POINTER] = TERMINAL_NAME [0];
;  242          1103  3                    CH$MOVE (%CHARCOUNT ('SYS$INPUT'), CH$PTR (UPLIT ('SYS$INPUT')), TERMINAL_NAME [0]);
;  243          1104  3
;  244          1105  3                    STATUS = LIB$GET_EF (EVENT_FLAG);
;  245          1106  3                    IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  246          1107  3
;  247        P 1108  3                    GETDVI_RESULT = $GETDVI (EFN = .EVENT_FLAG,
;  248        P 1109  3                                             DEVNAM = DEVNAM_DESC,
;  249          1110  3                                             ITMLST = DVI_ITEMS     );
;  250          1111  3
;  251          1112  3                    IF ( (NOT .GETDVI_RESULT) OR .DVI_RETURN_LENGTH EQL 0 )
;  252          1113  3                    THEN LIB$STOP (.GETDVI_RESULT);
;  253          1114  3
;  254          1115  3                    STATUS = LIB$FREE_EF (EVENT_FLAG);
;  255          1116  3                    IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  256          1117  3
;  257          1118  3            !+
;  258          1119  3            ! If SYS$INPUT is indeed a terminal device, go ahead and enable CTRL/C
;  259          1120  3            ! trapping to it.
;  260          1121  3            !-
;  261          1122  3                    IF .DEVICE_CLASS EQL DC$_TERM
;  262          1123  3                    THEN
;  263          1124  4                        BEGIN
;  264          1125  4            !+
;  265          1126  4            ! assign a channel to the terminal, if one doesn't already exist.
;  266          1127  4            !-
;  267          1128  4                        IF .TT_CHAN EQLU 0
;  268          1129  4                        THEN
;  269          1130  5                            BEGIN
;  270          1131  5                            ASSIGN_RESULT = $ASSIGN (DEVNAM = DEVNAM_DESC, CHAN = TT_CHAN);
;  271          1132  5
;  272          1133  6                            IF ( NOT .ASSIGN_RESULT)
;  273          1134  5                            THEN LIB$STOP (.ASSIGN_RESULT);
;  274          1135  4                            END;
;  275          1136  4
;  276          1137  4            !+
;  277          1138  4            ! issue the QIO enabling CTRL/C reception.
;  278          1139  4            !-
;  279          1140  4                        STATUS = LIB$GET_EF (EVENT_FLAG);
;  280          1141  4                        IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  281          1142  4
;  282        P 1143  4                        QIO_RESULT = $QIOW (EFN = .EVENT_FLAG,
;  283        P 1144  4                                            CHAN = .TT_CHAN,
```

```
;  284        P 1145  4                                FUNC = (IO$_SETMODE OR IO$M_OUTBAND OR IO$M_TT_ABORT),
;  285        P 1146  4                                P1 = CONTROL_C,
;  286          1147  4                                P2 = CONTROL_CHARS);
;  287          1148  4
;  288          1149  5              IF ( NOT .QIO_RESULT)
;  289          1150  4              THEN LIB$STOP (.QIO_RESULT);
;  290          1151  4
;  291          1152  4              STATUS = LIB$FREE_EF (EVENT_FLAG);
;  292          1153  4              IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  293          1154  4
;  294          1155  4 !+
;  295          1156  4 ! indicate CTRL/C recption is now enabled.
;  296          1157  4 !-
;  297          1158  4              CC_REALLY_ENABLED = 1;
;  298          1159  4              END
;  299          1160  4
;  300          1161  3          ELSE
;  301          1162  4              BEGIN
;  302          1163  4 !+
;  303          1164  4 ! otherwise, see if the process owns a terminal at all.
;  304          1165  4 !-
;  305          1166  4              STATUS = LIB$GET_EF (EVENT_FLAG);
;  306          1167  4              IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  307          1168  4
;  308        P 1169  4              GETJPI_RESULT = $GETJPI (EFN = .EVENT_FLAG,
;  309          1170  4                                       ITMLST = JPI_ITEMS );
;  310          1171  4
;  311          1172  5              IF (NOT .GETJPI_RESULT)
;  312          1173  4              THEN LIB$STOP (.GETJPI_RESULT);
;  313          1174  4
;  314          1175  4              STATUS = LIB$FREE_EF (EVENT_FLAG);
;  315          1176  4              IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  316          1177  4
;  317          1178  4 !+
;  318          1179  4 ! if so, enable CTRL/C reception to that terminal.  Otherwise, we cannot
;  319          1180  4 ! enable CTRL/C reception.
;  320          1181  4 !-
;  321          1182  4              IF .JPI_RETURN_LENGTH NEQ 0
;  322          1183  4              THEN
;  323          1184  5                  BEGIN
;  324          1185  5                  DEVNAM_DESC [DSC$W_LENGTH] = CH$FIND_CH ( 256,
;  325          1186  5                                                            CH$PTR (TERMINAL_NAME),
;  326          1187  5                                                            '') -
;  327          1188  5                                                    CH$PTR (TERMINAL_NAME);
;  328          1189  5                  DEVNAM_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  329          1190  5                  DEVNAM_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  330          1191  5                  DEVNAM_DESC [DSC$A_POINTER] = TERMINAL_NAME [0];
;  331          1192  5
;  332          1193  5 !+
;  333          1194  5 ! assign a channel to the terminal, if one doesn't already exist.
;  334          1195  5 !-
;  335          1196  5                  IF .TT_CHAN EQLU 0
;  336          1197  5                  THEN
;  337          1198  6                      BEGIN
;  338          1199  6                      ASSIGN_RESULT = $ASSIGN (DEVNAM = DEVNAM_DESC, CHAN = TT_CHAN);
;  339          1200  6
;  340          1201  7                      IF ( NOT .ASSIGN_RESULT)
```

```
;  341          1202  6                    THEN LIB$STOP (.ASSIGN_RESULT);
;  342          1203  5                    END;
;  343          1204  5
;  344          1205  5                !+
;  345          1206  5                ! issue the QIO enabling CTRL/C reception.
;  346          1207  5                !-
;  347          1208  5                    STATUS = LIB$GET_EF (EVENT_FLAG);
;  348          1209  5                    IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  349          1210  5
;  350        P 1211  5                    QIO_RESULT = $QIOW (EFN = .EVENT_FLAG,
;  351        P 1212  5                                        CHAN = .TT_CHAN,
;  352        P 1213  5                                        FUNC = (IO$_SETMODE OR IO$M_OUTBAND OR IO$M_TT_ABORT),
;  353        P 1214  5                                        P1 = CONTROL_C,
;  354          1215  5                                        P2 = CONTROL_CHARS);
;  355          1216  5
;  356          1217  6                    IF ( NOT .QIO_RESULT)
;  357          1218  5                    THEN LIB$STOP (.QIO_RESULT);
;  358          1219  5
;  359          1220  5                    STATUS = LIB$FREE_EF (EVENT_FLAG);
;  360          1221  5                    IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  361          1222  5
;  362          1223  5                !+
;  363          1224  5                ! indicate CTRL/C recption is now enabled.
;  364          1225  5                !-
;  365          1226  5                    CC_REALLY_ENABLED = 1;
;  366          1227  4                    END;
;  367          1228  4
;  368          1229  3                END;        ! Else
;  369          1230  3
;  370          1231  2            END;        ! If not CC_REALLY_ENABLED
;  371          1232  2
;  372          1233  2            !+
;  373          1234  2            ! indicate the CTRL/C reception is now enabled from the point of view
;  374          1235  2            ! of the user.
;  375          1236  2            !-
;  376          1237  2                CC_ENABLED_USER_PT_OF_VIEW = 1;
;  377          1238  2
;  378          1239  2            !+
;  379          1240  2            ! the CTRLC function always returns zero.
;  380          1241  2            !-
;  381          1242  2                RETURN (0);
;  382          1243  1                END;                                    ! end of BAS$CTRLC


                                                        .TITLE  BAS$CTRLC
                                                        .IDENT  \2-005\

                                                        .PSECT  _BAS$DATA,NOEXE,  PIC,2

                                    0000  00000 TT_CHAN:.WORD   0                                                   ;
                                      00  00002 RUN_CMD:.BYTE   0                                                   ;
                                      00  00003 CC_REALLY_ENABLED:
                                                        .BYTE   0                                                   ;
                                      00  00004 CC_ENABLED_USER_PT_OF_VIEW:
                                                        .BYTE   0                                                   ;

                                                        .PSECT  _BAS$CODE,NOWRT,  SHR,  PIC,2
```

```
                                00040004  00000 P.AAA:  .LONG   262148                                          ;
                                00000000  00004         .LONG   0                                             ;
                       00000000 00000000  00008         .LONG   0, 0                                          ;
                                031D0100  00010 P.AAB:  .LONG   52232448                                      ;
                                00000000  00014         .LONG   0                                             ;
                       00000000 00000000  00018         .LONG   0, 0                                          ;
   00  00  00  54  55  50  4E  49  24  53  59  53  00020 P.AAC:  .ASCII  \SYS$INPUT\<0><0><0>                   ;

                                                         .EXTRN  LIB$GET_EF, LIB$FREE_EF
                                                         .EXTRN  LIB$SIGNAL, LIB$STOP
                                                         .EXTRN  LIB$MATCH_COND, BAS$$CB_PUSH
                                                         .EXTRN  BAS$$CB_POP, BAS$$LINE
                                                         .EXTRN  BAS$$MODULE, BAS$HANDLER
                                                         .EXTRN  BAS$T_ERN, BAS$L_ERR
                                                         .EXTRN  BAS$L_ERL, OTS$$V_IOINPROG
                                                         .EXTRN  BAS$K_PROC_TRA
                                                         .EXTRN  BAS$_PROC_TRA, SYS$GETDVI
                                                         .EXTRN  SYS$ASSIGN, SYS$QIOW
                                                         .EXTRN  SYS$GETJPI

                                      0FFC 00000         .ENTRY  BAS$CTRLC, Save R2,R3,R4,R5,R6,R7,R8,R9,-     ; 1031
                                                                 R10,R11                                       ;
                  5B 00000000G   00   9E 00002         MOVAB   SYS$QIOW, R11                                 ;
                  5A 00000000G   00   9E 00009         MOVAB   SYS$ASSIGN, R10                               ;
                  59 00000000G   00   9E 00010         MOVAB   LIB$FREE_EF, R9                               ;
                  58 00000000G   00   9E 00017         MOVAB   LIB$GET_EF, R8                                ;
                  57 00000000'   EF   9E 0001E         MOVAB   TT_CHAN, R7                                   ;
                  56 00000000G   00   9E 00025         MOVAB   LIB$STOP, R6                                  ;
                  5E      FEC0   CE   9E 0002C         MOVAB   -320(SP), SP                                  ;
                  03        03   A7   E9 00031         BLBC    CC_REALLY_ENABLED, 1$                         ; 1066
                            01BB 31   00035         BRW     21$                                           ;
                         F8   AD   D4 00038 1$:      CLRL    CONTROL_CHARS                                 ; 1068
                FC   AD          08   D0 0003B         MOVL    #8, CONTROL_CHARS+4                           ;
                                 6E   7C 0003F         CLRQ    DEVICE_CLASS                                  ;
   E0   AD      8F   AF         10   28 00041         MOVC3   #16, P.AAA, DVI_ITEMS                         ; 1087
                E4   AD         6E   9E 00047         MOVAB   DEVICE_CLASS, DVI_ITEMS+4                     ; 1068
                E8   AD      04 AE   9E 0004B         MOVAB   DVI_RETURN_LENGTH, DVI_ITEMS+8                ;
                             08 AE   D4 00050         CLRL    JPI_RETURN_LENGTH                             ; 1087
   10   AE      8D   AF         10   28 00053         MOVC3   #16, P.AAB, JPI_ITEMS                         ; 1094
                14   AE      20 AE   9E 00059         MOVAB   TERMINAL_NAME, JPI_ITEMS+4                    ; 1087
                18   AE      08 AE   9E 0005E         MOVAB   JPI_RETURN_LENGTH, JPI_ITEMS+8                ;
                F0   AD 010E0009 8F   D0 00063         MOVL    #17694729, DEVNAM_DESC                        ; 1099
                F4   AD      20 AE   9E 0006B         MOVAB   TERMINAL_NAME, DEVNAM_DESC+4                  ; 1102
   20   AE      80   AF         09   28 00070         MOVC3   #9, P.AAC, TERMINAL_NAME                      ; 1103
                             0C AE   9F 00076         PUSHAB  EVENT_FLAG                                    ; 1105
                68           01   FB 00079         CALLS   #1, LIB$GET_EF                                ;
                52           50   D0 0007C         MOVL    R0, STATUS                                    ;
                05           52   E8 0007F         BLBS    STATUS, 2$                                    ; 1106
                             52   DD 00082         PUSHL   STATUS                                        ;
                66           01   FB 00084         CALLS   #1, LIB$STOP                                  ;
                             7E   7C 00087 2$:      CLRQ    -(SP)                                         ; 1110
                             7E   7C 00089         CLRQ    -(SP)                                         ;
                         E0  AD   9F 0008B         PUSHAB  DVI_ITEMS                                     ;
                         F0  AD   9F 0008E         PUSHAB  DEVNAM_DESC                                   ;
                             7E   D4 00091         CLRL    -(SP)                                         ;
                         28  AE   DD 00093         PUSHL   EVENT_FLAG                                    ;
```

```
  00000000G  00          08  FB 00096        CALLS   #8, SYS$GETDVI                   ;
             05          50  E9 0009D        BLBC    GETDVI_RESULT, 3$                ; 1112
                   04    AE  D5 000A0        TSTL    DVI_RETURN_LENGTH                ;
                         05  12 000A3        BNEQ    4$                               ;
                         50  DD 000A5 3$:    PUSHL   GETDVI_RESULT                    ; 1113
             66          01  FB 000A7        CALLS   #1, LIB$STOP                     ;
                   0C    AE  9F 000AA 4$:    PUSHAB  EVENT_FLAG                       ; 1115
             69          01  FB 000AD        CALLS   #1, LIB$FREE_EF                  ;
             52          50  D0 000B0        MOVL    R0, STATUS                       ;
             05          52  E8 000B3        BLBS    STATUS, 5$                       ; 1116
                         52  DD 000B6        PUSHL   STATUS                           ;
             66          01  FB 000B8        CALLS   #1, LIB$STOP                     ;
  00000042   8F          6E  D1 000BB 5$:    CMPL    DEVICE_CLASS, #66                ; 1122
                         64  12 000C2        BNEQ    10$                              ;
                         67  B5 000C4        TSTW    TT_CHAN                          ; 1128
                         15  12 000C6        BNEQ    6$                               ;
                         7E  7C 000C8        CLRQ    -(SP)                            ; 1131
                         57  DD 000CA        PUSHL   R7                               ;
                   F0    AD  9F 000CC        PUSHAB  DEVNAM_DESC                      ;
             6A          04  FB 000CF        CALLS   #4, SYS$ASSIGN                   ;
             54          50  D0 000D2        MOVL    R0, ASSIGN_RESULT                ;
             05          54  E8 000D5        BLBS    ASSIGN_RESULT, 6$                ; 1133
                         54  DD 000D8        PUSHL   ASSIGN_RESULT                    ; 1134
             66          01  FB 000DA        CALLS   #1, LIB$STOP                     ;
                   0C    AE  9F 000DD 6$:    PUSHAB  EVENT_FLAG                       ; 1140
             68          01  FB 000E0        CALLS   #1, LIB$GET_EF                   ;
             52          50  D0 000E3        MOVL    R0, STATUS                       ;
             05          52  E8 000E6        BLBS    STATUS, 7$                       ; 1141
                         52  DD 000E9        PUSHL   STATUS                           ;
             66          01  FB 000EB        CALLS   #1, LIB$STOP                     ;
                         7E  7C 000EE 7$:    CLRQ    -(SP)                            ; 1147
                         7E  7C 000F0        CLRQ    -(SP)                            ;
                   F8    AD  9F 000F2        PUSHAB  CONTROL_CHARS                    ;
                 0000V   CF  9F 000F5        PUSHAB  CONTROL_C                        ;
                         7E  7C 000F9        CLRQ    -(SP)                            ;
                         7E  D4 000FB        CLRL    -(SP)                            ;
             7E    1423  8F  3C 000FD        MOVZWL  #5155, -(SP)                     ;
             7E          67  3C 00102        MOVZWL  TT_CHAN, -(SP)                   ;
                   38    AE  DD 00105        PUSHL   EVENT_FLAG                       ;
             6B          0C  FB 00108        CALLS   #12, SYS$QIOW                    ;
             53          50  D0 0010B        MOVL    R0, QIO_RESULT                   ;
             05          53  E8 0010E        BLBS    QIO_RESULT, 8$                   ; 1149
                         53  DD 00111        PUSHL   QIO_RESULT                       ; 1150
             66          01  FB 00113        CALLS   #1, LIB$STOP                     ;
                   0C    AE  9F 00116 8$:    PUSHAB  EVENT_FLAG                       ; 1152
             69          01  FB 00119        CALLS   #1, LIB$FREE_EF                  ;
             52          50  D0 0011C        MOVL    R0, STATUS                       ;
             03          52  E9 0011F        BLBC    STATUS, 9$                       ; 1153
                       00CA  31 00122        BRW     20$                              ;
                       00C2  31 00125 9$:    BRW     19$                              ;
                   0C    AE  9F 00128 10$:   PUSHAB  EVENT_FLAG                       ; 1166
             68          01  FB 0012B        CALLS   #1, LIB$GET_EF                   ;
             52          50  D0 0012E        MOVL    R0, STATUS                       ;
             05          52  E8 00131        BLBS    STATUS, 11$                      ; 1167
                         52  DD 00134        PUSHL   STATUS                           ;
             66          01  FB 00136        CALLS   #1, LIB$STOP                     ;
                         7E  7C 00139 11$:   CLRQ    -(SP)                            ; 1170
```

```
                                   7E  D4 0013B        CLRL    -(SP)                                 ;
                        1C         AE  9F 0013D        PUSHAB  JPI_ITEMS                             ;
                                   7E  7C 00140        CLRQ    -(SP)                                 ;
                        24         AE  DD 00142        PUSHL   EVENT_FLAG                            ;
          00000000G  00            07  FB 00145        CALLS   #7, SYS$GETJPI                        ;
                     05            50  E8 0014C        BLBS    GETJPI_RESULT, 12$                    ; 1172
                                   50  DD 0014F        PUSHL   GETJPI_RESULT                         ; 1173
                     66            01  FB 00151        CALLS   #1, LIB$STOP                          ;
                        0C         AE  9F 00154 12$:   PUSHAB  EVENT_FLAG                            ; 1175
                     69            01  FB 00157        CALLS   #1, LIB$FREE_EF                       ;
                     52            50  D0 0015A        MOVL    R0, STATUS                            ;
                     05            52  E8 0015D        BLBS    STATUS, 13$                           ; 1176
                                   52  DD 00160        PUSHL   STATUS                                ;
                     66            01  FB 00162        CALLS   #1, LIB$STOP                          ;
                        08         AE  D5 00165 13$:   TSTL    JPI_RETURN_LENGTH                     ; 1182
                                   03  12 00168        BNEQ    14$                                   ;
                                 0086  31 0016A        BRW     21$                                   ;
 20   AE     0100    8F            00  3A 0016D 14$:   LOCC    #0, #256, TERMINAL_NAME               ; 1185
                                   02  12 00174        BNEQ    15$                                   ;
                                   51  D4 00176        CLRL    R1                                    ;
                     50     20     AE  9E 00178 15$:   MOVAB   TERMINAL_NAME, R0                     ; 1188
 F0   AD             51            50  A3 0017C        SUBW3   R0, R1, DEVNAM_DESC                   ;
                  F2 AD     010E   8F  B0 00181        MOVW    #270, DEVNAM_DESC+2                   ; 1189
                  F4 AD     20     AE  9E 00187        MOVAB   TERMINAL_NAME, DEVNAM_DESC+4          ; 1191
                                   67  B5 0018C        TSTW    TT_CHAN                               ; 1196
                                   15  12 0018E        BNEQ    16$                                   ;
                                   7E  7C 00190        CLRQ    -(SP)                                 ; 1199
                                   57  DD 00192        PUSHL   R7                                    ;
                        F0         AD  9F 00194        PUSHAB  DEVNAM_DESC                           ;
                     6A            04  FB 00197        CALLS   #4, SYS$ASSIGN                        ;
                     54            50  D0 0019A        MOVL    R0, ASSIGN_RESULT                     ;
                     05            54  E8 0019D        BLBS    ASSIGN_RESULT, 16$                    ; 1201
                                   54  DD 001A0        PUSHL   ASSIGN_RESULT                         ; 1202
                     66            01  FB 001A2        CALLS   #1, LIB$STOP                          ;
                        0C         AE  9F 001A5 16$:   PUSHAB  EVENT_FLAG                            ; 1208
                     68            01  FB 001A8        CALLS   #1, LIB$GET_EF                        ;
                     52            50  D0 001AB        MOVL    R0, STATUS                            ;
                     05            52  E8 001AE        BLBS    STATUS, 17$                           ; 1209
                                   52  DD 001B1        PUSHL   STATUS                                ;
                     66            01  FB 001B3        CALLS   #1, LIB$STOP                          ;
                                   7E  7C 001B6 17$:   CLRQ    -(SP)                                 ; 1215
                                   7E  7C 001B8        CLRQ    -(SP)                                 ;
                        F8         AD  9F 001BA        PUSHAB  CONTROL_CHARS                         ;
                     0000V         CF  9F 001BD        PUSHAB  CONTROL_C                             ;
                                   7E  7C 001C1        CLRQ    -(SP)                                 ;
                                   7E  D4 001C3        CLRL    -(SP)                                 ;
                     7E     1423   8F  3C 001C5        MOVZWL  #5155, -(SP)                          ;
                     7E            67  3C 001CA        MOVZWL  TT_CHAN, -(SP)                        ;
                        38         AE  DD 001CD        PUSHL   EVENT_FLAG                            ;
                     6B            0C  FB 001D0        CALLS   #12, SYS$QIOW                         ;
                     53            50  D0 001D3        MOVL    R0, QIO_RESULT                        ;
                     05            53  E8 001D6        BLBS    QIO_RESULT, 18$                       ; 1217
                                   53  DD 001D9        PUSHL   QIO_RESULT                            ; 1218
                     66            01  FB 001DB        CALLS   #1, LIB$STOP                          ;
                        0C         AE  9F 001DE 18$:   PUSHAB  EVENT_FLAG                            ; 1220
                     69            01  FB 001E1        CALLS   #1, LIB$FREE_EF                       ;
                     52            50  D0 001E4        MOVL    R0, STATUS                            ;
```

```
                          05        52  E8 001E7          BLBS    STATUS, 20$                             ; 1221
                                    52  DD 001EA 19$:      PUSHL   STATUS                                  ;
                          66        01  FB 001EC          CALLS   #1, LIB$STOP                            ;
                    03    A7        01  90 001EF 20$:      MOVB    #1, CC_REALLY_ENABLED                   ; 1226
                    04    A7        01  90 001F3 21$:      MOVB    #1, CC_ENABLED_USER_PT_OF_VIEW          ; 1237
                                    50  D4 001F7          CLRL    R0                                      ; 1242
                                        04 001F9          RET                                             ; 1243

; Routine Size:  506 bytes,    Routine Base:  _BAS$CODE + 002C

;  383          1244  1
```

```
;  385          1245  1 GLOBAL ROUTINE BAS$RCTRLC =                    ! Disable Control C interrupts
;  386          1246  1
;  387          1247  1 !++
;  388          1248  1 ! FUNCTIONAL DESCRIPTION:
;  389          1249  1 !
;  390          1250  1 !       Disable Control C traps, so that a Control C will cause the
;  391          1251  1 !       user's program to stop, as usual.
;  392          1252  1 !
;  393          1253  1 ! FORMAL PARAMETERS:
;  394          1254  1 !
;  395          1255  1 !       NONE
;  396          1256  1 !
;  397          1257  1 ! IMPLICIT INPUTS:
;  398          1258  1 !
;  399          1259  1 !       NONE
;  400          1260  1 !
;  401          1261  1 ! IMPLICIT OUTPUTS:
;  402          1262  1 !
;  403          1263  1 !       NONE
;  404          1264  1 !
;  405          1265  1 ! ROUTINE VALUE:
;  406          1266  1 ! COMPLETION CODES:
;  407          1267  1 !
;  408          1268  1 !       Always returns zero.
;  409          1269  1 !
;  410          1270  1 ! SIDE EFFECTS:
;  411          1271  1 !
;  412          1272  1 !       Leaves Control C traps disabled.
;  413          1273  1 !
;  414          1274  1 !--
;  415          1275  1
;  416          1276  2     BEGIN
;  417          1277  2
;  418          1278  2     LOCAL
;  419          1279  2         EVENT_FLAG,
;  420          1280  2         STATUS,
;  421          1281  2         QIO_RESULT;
;  422          1282  2
;  423          1283  2 !+
;  424          1284  2 ! Only turn CTRL/C reception off if it is currently on, and we're NOT in
;  425          1285  2 ! the environment (RUN_CMD).  CTRL/C reception should always be enabled
;  426          1286  2 ! (from the point of view of the user) when running in the environment.
;  427          1287  2 !-
;  428          1288  2
;  429          1289  2     IF ((.TT_CHAN NEQU 0) AND ( .CC_REALLY_ENABLED ))
;  430          1290  2     THEN
;  431          1291  3         BEGIN
;  432          1292  3 !+
;  433          1293  3 ! If we are in the RUN command (where control Cs should always remain
;  434          1294  3 ! enabled) or if control Cs are not enabled, don't issue the QIO.
;  435          1295  3 !-
;  436          1296  3
;  437          1297  4         IF ( NOT .RUN_CMD)
;  438          1298  4         THEN
;  439          1299  5             BEGIN
;  440          1300  5
;  441          1301  5             STATUS = LIB$GET_EF (EVENT_FLAG);
```

```
;  442     1302  4              IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  443     1303  4
;  444     1304  4          !+
;  445     1305  4          ! We disable reception of CTRL/C ASTs by issuing a $CANCEL on the channel.
;  446     1306  4          !-
;  447     1307  4              QIO_RESULT = $CANCEL ( CHAN = .TT_CHAN);
;  448     1308  4
;  449     1309  4              IF ( NOT .QIO_RESULT) THEN LIB$STOP (.QIO_RESULT);
;  450     1310  4
;  451     1311  4              STATUS = LIB$FREE_EF (EVENT_FLAG);
;  452     1312  4              IF (NOT .STATUS) THEN LIB$STOP (.STATUS);
;  453     1313  4
;  454     1314  4              CC_REALLY_ENABLED = 0;
;  455     1315  3              END;
;  456     1316  3
;  457     1317  3          !+
;  458     1318  3          ! Indicate that the user does not want control C traps.
;  459     1319  3          !-
;  460     1320  2          END;
;  461     1321  2
;  462     1322  2      CC_ENABLED_USER_PT_OF_VIEW = 0;
;  463     1323  2
;  464     1324  2      RETURN (0);
;  465     1325  1      END;                                    ! end of BAS$RCTRLC


                                                .EXTRN  SYS$CANCEL

                                001C 00000      .ENTRY  BAS$RCTRLC, Save R2,R3,R4       ; 1245
                    54 00000000G  00   9E 00002 MOVAB   LIB$STOP, R4
                    53 00000000'  EF   9E 00009 MOVAB   TT_CHAN, R3
                    5E            04   C2 00010 SUBL2   #4, SP
                                  63   B5 00013 TSTW    TT_CHAN                     ; 1289
                                  45   13 00015 BEQL    4$
                    41      03    A3   E9 00017 BLBC    CC_REALLY_ENABLED, 4$
                    3D      02    A3   E8 0001B BLBS    RUN_CMD, 4$                 ; 1297
                                  5E   DD 0001F PUSHL   SP                          ; 1301
          00000000G 00            01   FB 00021 CALLS   #1, LIB$GET_EF
                    52            50   D0 00028 MOVL    R0, STATUS
                    05            52   E8 0002B BLBS    STATUS, 1$                  ; 1302
                                  52   DD 0002E PUSHL   STATUS
                    64            01   FB 00030 CALLS   #1, LIB$STOP
                    7E            63   3C 00033 1$:     MOVZWL  TT_CHAN, -(SP)      ; 1307
          00000000G 00            01   FB 00036 CALLS   #1, SYS$CANCEL
                    05            50   E8 0003D BLBS    QIO_RESULT, 2$              ; 1309
                                  50   DD 00040 PUSHL   QIO_RESULT
                    64            01   FB 00042 CALLS   #1, LIB$STOP
                                  5E   DD 00045 2$:     PUSHL   SP                  ; 1311
          00000000G 00            01   FB 00047 CALLS   #1, LIB$FREE_EF
                    52            50   D0 0004E MOVL    R0, STATUS
                    05            52   E8 00051 BLBS    STATUS, 3$                  ; 1312
                                  52   DD 00054 PUSHL   STATUS
                    64            01   FB 00056 CALLS   #1, LIB$STOP
                          03      A3   94 00059 3$:     CLRB    CC_REALLY_ENABLED   ; 1314
                          04      A3   94 0005C 4$:     CLRB    CC_ENABLED_USER_PT_OF_VIEW  ; 1322
                                  50   D4 0005F CLRL    R0                          ; 1324
```

```
                                    04 00061        RET                                             ; 1325

; Routine Size:  98 bytes,    Routine Base:  _BAS$CODE + 0226

;  466          1326  1
```

```
;  468     1327  1 GLOBAL ROUTINE BAS$$CTRLC_INIT : NOVALUE =      ! Set up for RUN command
;  469     1328  1
;  470     1329  1 !++
;  471     1330  1 ! FUNCTIONAL DESCRIPTION:
;  472     1331  1 !
;  473     1332  1 !       Set up for the RUN environment.  Since this image is to run under the RUN
;  474     1333  1 !       command, control C traps are always enabled, from the point of view of
;  475     1334  1 !       VMS.  If one goes off when the user has not enabled for control C traps,
;  476     1335  1 !       the user is not allowed to intercept the signal (because of its severity)
;  477     1336  1 !       and the keyboard monitor gets it instead.
;  478     1337  1 !
;  479     1338  1 ! FORMAL PARAMETERS:
;  480     1339  1 !
;  481     1340  1 !       NONE
;  482     1341  1 !
;  483     1342  1 ! IMPLICIT INPUTS:
;  484     1343  1 !
;  485     1344  1 !       NONE
;  486     1345  1 !
;  487     1346  1 ! IMPLICIT OUTPUTS:
;  488     1347  1 !
;  489     1348  1 !       NONE
;  490     1349  1 !
;  491     1350  1 ! ROUTINE VALUE:
;  492     1351  1 ! COMPLETION CODES:
;  493     1352  1 !
;  494     1353  1 !       NONE
;  495     1354  1 !
;  496     1355  1 ! SIDE EFFECTS:
;  497     1356  1 !
;  498     1357  1 !       Leaves Control C traps disabled from the user's point of view, but
;  499     1358  1 !       enabled from VMS's point of view.
;  500     1359  1 !
;  501     1360  1 !--
;  502     1361  1
;  503     1362  2     BEGIN
;  504     1363  2 !+
;  505     1364  2 ! Make sure the $ASSIGN and $QIO have been done.
;  506     1365  2 !-
;  507     1366  2     BAS$CTRLC ();
;  508     1367  2 !+
;  509     1368  2 ! Flag that we are in the RUN environment.  This will prevent the
;  510     1369  2 ! Control C enable from being turned off, from the point of view
;  511     1370  2 ! of VMS.
;  512     1371  2 !-
;  513     1372  2     RUN_CMD = 1;
;  514     1373  2 !+
;  515     1374  2 ! Turn control C enable off from the user's point of view.
;  516     1375  2 !-
;  517     1376  2     BAS$RCTRLC ();
;  518     1377  2     RETURN;
;  519     1378  1     END;                                        ! end of BAS$$CTRLC_INIT
```

```
                                    0000 00000        .ENTRY  BAS$$CTRLC_INIT, Save nothing                ; 1327
                          FD9D   CF          00  FB 00002        CALLS   #0, BAS$CTRLC                        ; 1366
                00000000'  EF          01  90 00007        MOVB    #1, RUN_CMD                          ; 1372
                        8C   AF          00  FB 0000E        CALLS   #0, BAS$RCTRLC                       ; 1376
                                            04 00012        RET                                          ; 1378

; Routine Size:  19 bytes,    Routine Base:  _BAS$CODE + 0288

;  520          1379  1
```

```
;   522          1380  1 GLOBAL ROUTINE BAS$$SIGNAL_CTRLC : NOVALUE =    ! Signal CTRL/C
;   523          1381  1
;   524          1382  1 !++
;   525          1383  1 ! FUNCTIONAL DESCRIPTION:
;   526          1384  1 !
;   527          1385  1 !       Signals CTRL/C to the BASIC program.
;   528          1386  1 !
;   529          1387  1 ! FORMAL PARAMETERS:
;   530          1388  1 !
;   531          1389  1 !       NONE
;   532          1390  1 !
;   533          1391  1 ! IMPLICIT INPUTS:
;   534          1392  1 !
;   535          1393  1 !       NONE
;   536          1394  1 !
;   537          1395  1 ! IMPLICIT OUTPUTS:
;   538          1396  1 !
;   539          1397  1 !       NONE
;   540          1398  1 !
;   541          1399  1 ! ROUTINE VALUE:
;   542          1400  1 ! COMPLETION CODES:
;   543          1401  1 !
;   544          1402  1 !       NONE
;   545          1403  1 !
;   546          1404  1 ! SIDE EFFECTS:
;   547          1405  1 !
;   548          1406  1 !       Calls the user's code by Signaling.
;   549          1407  1 !       If the user is not enabled (which means that the program must
;   550          1408  1 !       be being run under the RUN command) then the signal goes to
;   551          1409  1 !       the keyboard monitor, which may do a continue or an unwind.
;   552          1410  1 !
;   553          1411  1 !--
;   554          1412  1
;   555          1413  2     BEGIN
;   556          1414  2
;   557          1415  2     LOCAL
;   558          1416  2         COND_VAL : BLOCK [4, BYTE],
;   559          1417  2         FMP :    REF BLOCK [0,BYTE] FIELD (BSF$FCD),
;   560          1418  2         MOD_NAME_ADDR;
;   561          1419  2
;   562          1420  2     BUILTIN
;   563          1421  2         FP;
;   564          1422  2
;   565          1423  2 !+
;   566          1424  2 ! if we're not really enabled, don't bother signalling.
;   567          1425  2 !-
;   568          1426  2     IF NOT .CC_REALLY_ENABLED
;   569          1427  2     THEN
;   570          1428  2         RETURN;
;   571          1429  2
;   572          1430  2 !+
;   573          1431  2 ! Search for a BASIC major frame.
;   574          1432  2 !-
;   575          1433  2     FMP = .FP;
;   576          1434  2
;   577          1435  3     WHILE ( (.FMP NEQ 0) AND (.FMP [BSF$A_HANDLER] NEQA BAS$HANDLER) )
;   578          1436  2     DO
```

```
;  579          1437  3          BEGIN
;  580          1438  3          FMP = .FMP [BSF$A_SAVED_FP];
;  581          1439  3          END;
;  582          1440  2
;  583          1441  2      !+
;  584          1442  2      ! get current error line (ERL) and error module (ERN$), and define current
;  585          1443  2      ! error as "Programmable ^C trap".
;  586          1444  2      !-
;  587          1445  2          IF (.FMP NEQ 0)
;  588          1446  2          THEN
;  589          1447  2              BEGIN
;  590          1448  2              BAS$L_ERL = BAS$$LINE (.FMP);
;  591          1449  2              MOD_NAME_ADDR = BAS$$MODULE (.FMP);
;  592          1450  2              BAS$T_ERN [DSC$A_POINTER] = .MOD_NAME_ADDR + 1;
;  593          1451  2              BAS$T_ERN [DSC$W_LENGTH] = .BLOCK [.MOD_NAME_ADDR, 0, 0, 8, 0; 1, BYTE];
;  594          1452  2              BAS$T_ERN [DSC$B_CLASS] = DSC$K_CLASS_S;
;  595          1453  2              BAS$T_ERN [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  596          1454  2              BAS$L_ERR = BAS$K_PROC__TRA;
;  597          1455  2              END;
;  598          1456  2
;  599          1457  2      !+
;  600          1458  2      ! Now signal the appropriate BASIC condition for Control C.  By default, the
;  601          1459  2      ! severity for CTRL/C is ERROR.  If the user is not enabled, signal information.
;  602          1460  2      ! BAS$HANDLER will gain control when the exception is signalled, and check the
;  603          1461  2      ! severity.  If it is ERROR, then the assumption is that the user has a handler
;  604          1462  2      ! for CTRL/C and the user's handler is called.  Otherwise (informational),
;  605          1463  2      ! control will be returned to KMON (environment) or DCL (run from DCL).
;  606          1464  2      !-
;  607          1465  2          COND_VAL = BAS$_PROC__TRA;
;  608          1466  2
;  609          1467  2          IF ( NOT .CC_ENABLED_USER_PT_OF_VIEW)
;  610          1468  2          THEN COND_VAL [STS$V_SEVERITY] = STS$K_INFO;
;  611          1469  2
;  612          1470  2          LIB$SIGNAL (.COND_VAL);
;  613          1471  2
;  614          1472  2      !+
;  615          1473  2      ! If we get to here, then the program was being run from the environment, the
;  616          1474  2      ! user had no CTRL/C handler, and the keyboard monitor received the CONTINUE
;  617          1475  2      ! command.  Dismiss the AST (done automatically by returning).
;  618          1476  2      !-
;  619          1477  2          RETURN;
;  620          1478  1          END;                                    ! end of BAS$$SIGNAL_CTRLC


                                  000C 00000          .ENTRY  BAS$$SIGNAL_CTRLC, Save R2,R3                   ; 1380
53 00000000G   00   9E 00002          MOVAB   BAS$T_ERN+4, R3                                                 ;
69 00000000'   EF   E9 00009          BLBC    CC_REALLY_ENABLED, 5$                                           ; 1426
52             5D   D0 00010          MOVL    FP, FMP                                                         ; 1433
               12   13 00013 1$:      BEQL    2$                                                              ; 1435
50 00000000G   00   9E 00015          MOVAB   BAS$HANDLER, R0                                                 ;
50             62   D1 0001C          CMPL    (FMP), R0                                                       ;
               06   13 0001F          BEQL    2$                                                              ;
52       0C    A2   D0 00021          MOVL    12(FMP), FMP                                                    ; 1438
               EC   11 00025          BRB     1$                                                              ; 1435
```

```
                                        52  D5 00027 2$:    TSTL    FMP                                              ; 1445
                                        32  13 00029        BEQL    3$                                               ;
                                        52  DD 0002B        PUSHL   FMP                                              ; 1448
                 00000000G  00          01  FB 0002D        CALLS   #1, BAS$$LINE                                    ;
                 00000000G  00          50  D0 00034        MOVL    R0, BAS$L_ERL                                    ;
                                        52  DD 0003B        PUSHL   FMP                                              ; 1449
                 00000000G  00          01  FB 0003D        CALLS   #1, BAS$$MODULE                                  ;
                            63      01  A0  9E 00044        MOVAB   1(R0), BAS$T_ERN+4                               ; 1450
                        FC  A3          60  9B 00048        MOVZBW  (MOD_NAME_ADDR), BAS$T_ERN                       ; 1451
                        FE  A3    010E  8F  B0 0004C        MOVW    #270, BAS$T_ERN+2                                ; 1453
                 00000000G  00 00000000G 8F D0 00052        MOVL    #BAS$K_PROC__TRA, BAS$L_ERR                      ; 1454
                            50 00000000G 8F D0 0005D 3$:    MOVL    #BAS$_PROC__TRA, COND_VAL                        ; 1465
                            05 00000000' EF E8 00064        BLBS    CC_ENABLED_USER_PT_OF_VIEW, 4$                   ; 1467
        50            03    00          03  F0 0006B        INSV    #3, #0, #3, COND_VAL                             ; 1468
                                        50  DD 00070 4$:    PUSHL   COND_VAL                                         ; 1470
                 00000000G  00          01  FB 00072        CALLS   #1, LIB$SIGNAL                                   ;
                                            04 00079 5$:    RET                                                      ; 1478

; Routine Size:  122 bytes,    Routine Base:  _BAS$CODE + 029B

;  621          1479  1
```

```
;  623          1480  1 ROUTINE CONTROL_C : NOVALUE =                    ! Handle a Control C interrupt
;  624          1481  1
;  625          1482  1 !++
;  626          1483  1 ! FUNCTIONAL DESCRIPTION:
;  627          1484  1 !
;  628          1485  1 !       This is the RTL AST routine for CTRL/C's deliered to BASIC programs.
;  629          1486  1 !       It handles the Control C interrupt, and may signal it to the BASIC
;  630          1487  1 !       program, depending on whether I/O was interrupted or not.
;  631          1488  1 !
;  632          1489  1 ! FORMAL PARAMETERS:
;  633          1490  1 !
;  634          1491  1 !       NONE
;  635          1492  1 !
;  636          1493  1 ! IMPLICIT INPUTS:
;  637          1494  1 !
;  638          1495  1 !       NONE
;  639          1496  1 !
;  640          1497  1 ! IMPLICIT OUTPUTS:
;  641          1498  1 !
;  642          1499  1 !       NONE
;  643          1500  1 !
;  644          1501  1 ! ROUTINE VALUE:
;  645          1502  1 ! COMPLETION CODES:
;  646          1503  1 !
;  647          1504  1 !       NONE
;  648          1505  1 !
;  649          1506  1 ! SIDE EFFECTS:
;  650          1507  1 !
;  651          1508  1 !       May call the user's code by Signaling.
;  652          1509  1 !
;  653          1510  1 !--
;  654          1511  1
;  655          1512  2     BEGIN
;  656          1513  2
;  657          1514  2     GLOBAL REGISTER
;  658          1515  2         CCB = K_CCB_REG : REF BLOCK [, BYTE];
;  659          1516  2
;  660          1517  2     LOCAL
;  661          1518  2         COND_VAL : BLOCK [4, BYTE],
;  662          1519  2         FMP :    REF BLOCK [0,BYTE] FIELD (BSF$FCD),
;  663          1520  2         MOD_NAME_ADDR;
;  664          1521  2
;  665          1522  2     BUILTIN
;  666          1523  2         FP;
;  667          1524  2
;  668          1525  2 !+
;  669          1526  2 ! search for I/O active; if I/O is active on any channel then assume
;  670          1527  2 ! this AST interrupted it.
;  671          1528  2 !-
;  672          1529  2     INCR LUN FROM 0 TO LUB$K_LUN_MAX DO
;  673          1530  3         BEGIN
;  674          1531  4             IF ( .OTS$$V_IOINPROG [.LUN] NEQU 0 )
;  675          1532  3             THEN
;  676          1533  3                 !+
;  677          1534  3                 ! I/O is active.  Push the channel and see if this is a
;  678          1535  3                 ! forcible (i.e., terminal) device.
;  679          1536  3                 !-
```

```
;  680          1537  4                BEGIN
;  681          1538  4                BAS$$CB_PUSH ( .LUN + LUB$K_ILUN_MIN, LUB$K_ILUN_MIN );
;  682          1539  4                IF .CCB [LUB$V_FORCIBLE]
;  683          1540  4                THEN
;  684          1541  4                    !+
;  685          1542  4                    ! this is indeed a terminal device.  pop this channel and
;  686          1543  4                    ! return.  the record level routines will notice the
;  687          1544  4                    ! RMS$_CONTROLC return status and signal.
;  688          1545  4                    !
;  689          1546  4                    ! note that returning dismisses the AST.
;  690          1547  4                    !-
;  691          1548  5                    BEGIN
;  692          1549  5                        BAS$$CB_POP ();
;  693          1550  5                        RETURN;
;  694          1551  4                    END;
;  695          1552  4
;  696          1553  4                !+
;  697          1554  4                ! not a terminal device on this channel.  pop the channel
;  698          1555  4                ! and continue looking.
;  699          1556  4                !-
;  700          1557  4                BAS$$CB_POP ();
;  701          1558  4
;  702          1559  3                END;
;  703          1560  2            END;
;  704          1561  2
;  705          1562  2        !+
;  706          1563  2        ! An I/O was not interrupted, or I/O to a device other than a terminal was
;  707          1564  2        ! interrupted.  Signal the CTRLC condition at this time.
;  708          1565  2        !-
;  709          1566  2            BAS$$SIGNAL_CTRLC();
;  710          1567  2
;  711          1568  2            RETURN;
;  712          1569  1            END;                                    ! end of CONTROL_C


                                                081C 00000 CONTROL_C:
                                                           .WORD   Save R2,R3,R4,R11                   ; 1480
                                    54 00000000G   00   9E 00002           MOVAB   BAS$$CB_POP, R4
                                                   53   D4 00009           CLRL    LUN                 ; 1529
          50 00000000G   00           01                53   EF 0000B 1$:      EXTZV   LUN, #1, OTS$$V_IOINPROG, R0 ; 1531
                                                   50   D5 00014           TSTL    R0
                                                   17   13 00016           BEQL    3$
                                    52         F8     A3   9E 00018           MOVAB   -8(LUN), R2         ; 1538
                                    50                08   CE 0001C           MNEGL   #8, R0
                                          00000000G   00   16 0001F           JSB     BAS$$CB_PUSH
                        03          FE      AB             06   E1 00025           BBC     #6, -2(CCB), 2$ ; 1539
                                                   64   16 0002A           JSB     BAS$$CB_POP         ; 1549
                                                        04 0002C           RET                         ; 1548
                                                   64   16 0002D 2$:       JSB     BAS$$CB_POP         ; 1557
                        D4                  53 00000077    8F   F3 0002F 3$:       AOBLEQ  #119, LUN, 1$ ; 1529
                                  FF4A      CF             00   FB 00037           CALLS   #0, BAS$$SIGNAL_CTRLC ; 1566
                                                        04 0003C           RET                         ; 1569

; Routine Size:  61 bytes,    Routine Base:  _BAS$CODE + 0315
```

```
;  713          1570  1
;  714          1571  1 END                                  ! end of module BAS$CTRLC
;  715          1572  0 ELUDOM
```

```
;                                PSECT SUMMARY

;       Name                    Bytes                   Attributes

;  _BAS$DATA                        5  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _BAS$CODE                      850  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
;                            Library Statistics

;                                    -------- Symbols --------      Pages       Processing
;       File                         Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]STARLET.L32;1   9776        21         0         581        00:01.1
```

```
;                                COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:BASCTRLC/OBJ=OBJ$:BASCTRLC MSRC$:BASCTRLC/UPDATE=(ENH$:BASCTRLC)

; Size:          806 code + 49 data bytes
; Run Time:          00:19.6
; Elapsed Time:      00:43.2
; Lines/CPU Min:      4802
; Lexemes/CPU-Min: 26578
; Memory Used:  220 pages
; Compilation Complete
```

BASCLOSE
LIS

BASCONCAT
LIS

BASCTRLO
LIS

BASCHANGE
LIS

BASCTRLC
LIS

BASCHAIN
LIS

BASCOPYFD
LIS

BASCHR
LIS

BASCMPAPP
LIS

BASCVTOUT
LIS

BASCOPOS
LIS